International Journal of Engineering Research (ISSN : 2319-6890
Volume No.2, Issue No.7, pp : 456-461 01 Nov 2013

Scalable Content Management System

Sandeep Krishna S, Jayant Dani

Product Technology Group, CEG-SEG, Tata Consultancy Services Ltd., Mumbai., India

sandeepkrishna.s@tcs.com, jayant.dani@tcs.com

Abstract—***Immense growth in the volume of contents every day demands more scalable system to handle and overcome difficulties in capture, storage, transform, search, sharing and visualization of data, where the data can be a structured or unstructured data of any type. A system to manage the growing contents and overcome the issues and complexity faced using appropriate technologies would advantage over measurable qualities like flexibility, interoperability, customizability, security, auditability, quality, community support, options and cost of licensing. So architecting aContent Management System in terms of enterprise needs and a scalable solution to manage the huge data growth necessitates a Scalable Content Management System.***

Keywords—Content Management System, Open Source, Product Line Approach, Scalability, Big Data Store Plugin, Benchmarking Research.

1. Introduction

A Scalable System is requiredto handle the growing volume of data and perform well, to meet the user needs in terms of performance, concurrency and load. In Software, scalability is based upon the number of users or load generatorsincremented on a fixed hardware configuration. Hardware Scalability is based upon the number of physical processors incremented in the hardware configuration, while keeping the user load per processor fixed. Meanwhile Horizontal Scaling is provided to scale out by adding more nodes to a system, such as adding more servers to an application to distribute the load.While Vertical Scaling scale's up by adding resources to a single node, such as addition of memory to a single machine. Hence a Scalable Content Management System (CMS) is built considering these scalability factors.

CMS is a system which manages the information or data that is specific to a website or application or data repository. The CMS allows the content to create, edit, version, publish, search and maintain from a central interface. CMS is a technology to manage the content life cycle irrespective of type of data. Various types of CMS are designed based on the enterprise requirements like Web Content Management, Document Management, Digital Asset Management, Records Management, Information Rights Management and Email Management.

To build this system Open Source technologies are considered as they are more powerful than most of the commercial products because it harnesses the brilliance of the people around the world who are contributing to build something amazing regardless of distance, origin, language and culture. It makes business sense apart from being free it is backed by hundreds of companies around the world rather than one company owning all its rights. It is independent, easy to manage, continuous real time improvement and hands on exploration is the significance of open source technologies.

2. Technology Research

An analysis of enterprise and open source CMS products and frameworks are identified and analysed. The products identified are Alfresco, Magnolia, Hippo and Adobe CQ5. These products are developed using Apache Jackrabbit (AJR) a fully conforming implementation of the Content Repository using Java Technology and developed over the Java Content Repository 2.0 specifications JSR 170 and JSR 283.

AJR based on Apache License 2.0 is a liberal license and allows modification to source code and packaging it in product without any licensing/legal obligations, hence it is selected to provide core CMS engine features.

A research on supportingCMS components are done based on the enterprise requirements to build a enterprise specific CMS. Some of the key components identified and technology research on selecting them are as follows,

- Content Categorisation
- Content Security
- Workflow Manager for content life cycle
- Template Engine
- Content Search
- Data Repository

Content Categorisation – The key services for a content management system is to maintain content hierarchy, content metadata's, version, workspace, and content relationship. Apache Jackrabbit supports in providing these services. The Content created can be of any type structured or unstructured and each content type has its own set of metadata's. The metadata's associated for a content type can be composite so that it can be modified as required and the version of the same has to be maintained. Apache Jackrabbit is fully conforming technology that supports the implementation of these features and modified as required.

Content Security – This feature is required to secure and protect your contents and provide access to the users based on the roles and privileges to access the CMS. Access Control List (ACLs) is provided to secure access to data repository and roles are created to secure access at interface level to the users accessing the system. The Apache Jackrabbit by default provides security and access manager. To provide a complete access manager with User Role Management, various external components like Spring Security, Java Authentication and Authorisation Service (JAAS) were considered. However a customised asset with required API's for User Role Management (URM) is developed using these technologies.

Workflow Manager – A key component for a CMS is to manage content lifecycle. On analysis of technologies to enable this feature JBPM provides a flexible Business Process Management (BPM) Suite. JBPM with Drools Guvnor provides a web designer to define the processes. As JBPM suite is a complete JBPM solution which exceeds the workflow requirement of CMS. Meanwhile Bizagi process modeller is also used to design the process which generates XPDL. Hence a customised Workflow Manager using EJB, Bizagi process and

velocity scripting for decision/routing logic is built and deployed in JBoss server which suffices all the requirements of a workflow to manage content lifecycle. The required API services are identified and builtto enable workflow in a CMS.

Template Engine – This is required to preview the contents in a particular format. Templates play an important role in case of Web Content Management where the templates are defined for the web pages and contents created are previewed via the template to provide a better visualization of the content and creation of web pages. Open source template engines like free marker, thymleaf or any jsp viewer are used to preview the content. On an analysis and need across various products the engine used to write and preview templates was found to be Freemarker and hence writing/previewing the templates using freemarker is set as standard and have options of using other formats like JSP and HTML can also be selected by providing the corresponding view resolver.

Content Search – Search of contents is another key feature. A content search is done using a content name/id. To get a refined search it can be done using the metadata's of content. Elastic or Full text search are searches that are performed on the file attachments like word, pdf, etc., by providing the keyword. Apache Lucene – Apache Solr are search engines arefast open source enterprise search platform.Its major features include powerful full-text search, hit highlighting, faceted search, near real-time indexing, dynamic clustering, database integration, rich document handling, and geospatial search. Based on the need the search is performed to obtain the required search results.

Data Repository–Data's are considered to be a relational database management system or a file system based repository based on requirement. As an open source system Postgres is considered to be one of the fully compliant freely available database system which can be configured when initializing a repository using JCR or JCR by itself has a default file system based repository which stores data in the repository and enable all the features related to data search, indexing, versioning, workspace creation and categorization of content.

3. Product Line Approach

The architecture of the product is created based on Product Line Approach. As product line approach is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. Also software product lines allows developers to take advantage of common characteristics among a family of products they produce and as a result increase quality, reduce cost and save time. It aims to create Reference Architecture first and then based on specific business needs independent Products are developed. Figure1 represents the product line approach.

Initially Product Line is built by creating Core Asset libraries and then wiring the assets to meet a business requirement.

Widely used industry design patterns were deployed for wiring of core asset libraries to come up with the base model of CMS, keeping in mind business demands of today's modern world and then products like Web Content Management, Digital Asset Management and Knowledge Management are developed on top of core CMS model.

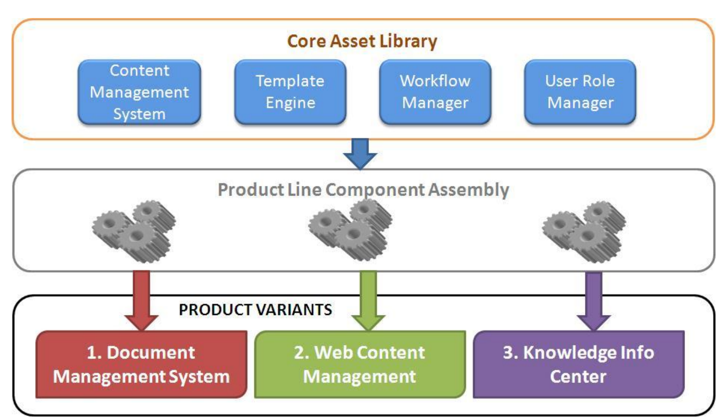


Figure1: Product Line Approach

The high level technical architecture for the CMS and the component wiring in to it based on the enterprise requirements is represented in Figure 2.

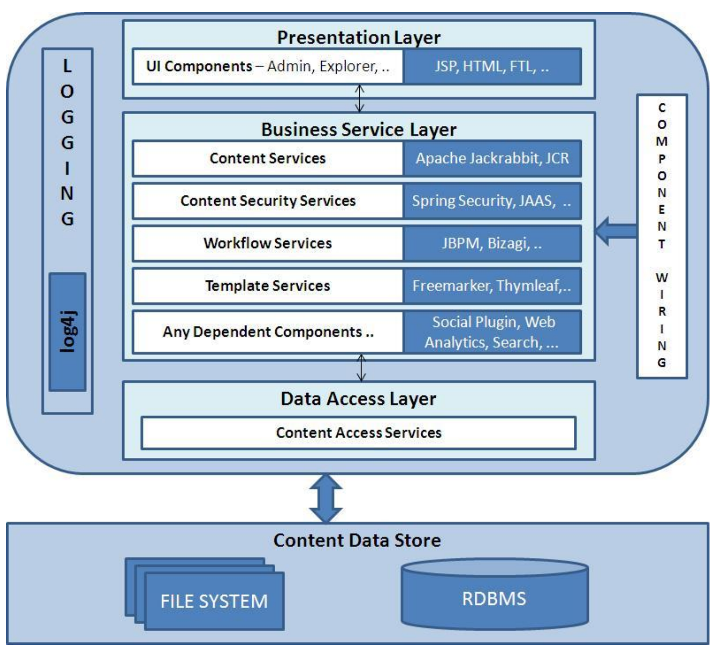


Figure2: High Level Technical Architecture

4. Content Scalability

Considering scalability for a Content Management System to handle the growing size of data or to handle a data's of huge sizes the scalability is measured on three main dimensions like volume, velocity and variety. The volume of the data can be in terabyte or petabyte with the velocity in which it is been created followed by the variety of data's which can be structured or unstructured.

The **Universal Scalability Law** (USL) is used to quantify the scalability of hardware or software systems it uses sparse measurements from an existing system to predict the throughput for different loads and can be used to learn more about the scalability limitations of the system. Refer Section 7[vii].

The Relative Capacity C (N) of a computational platform is given by,

$$C(N) = \frac{\quad}{1 + \alpha (N - 1) + \beta N (N - 1)}$$

Where N represents either the number of physical processors in the hardware configuration or the number of users driving the software application. The parameters α and β represent respectively the levels of contention (e.g., queuing for shared resources) and coherency delay (i.e., latency for data to become consistent) in the system. The parameter values are defined in the range: $0 \leq \alpha, \beta < 1$.

The USL is applied for,

- **Virtual Load Testing:** For load measurement of data to determine application scalability under larger user loads.
- **Number of Measurements:** A statistical regression using the USL model, at the best can only be referring to n samples of the total system throughput X where the random variables N or p are also changing. This is hardly the conventional statistical procedure for a performance analysis.
- **Performance Heuristics:** The relative sizes of the α and β parameters tells us, whether contention effects or coherency effects are responsible for poor scalability.
- **Performance Diagnostics:** The application developers, system architects may easily identify the problem once the USL is applied for α and β parameters and limits the diagnostic capability.
- **Production Environments:** Applying the USL to performance data collected from production environments with mixed workloads is analysed
- **Scalability Zones:** Letting the data fall across a number of regions whose boundaries are defined by a set of fitted scalability curves. The boundaries define zones which can be directly identified using some queuing effects and this can be immensely helpful to apps developer to tweak their code to improve performance.

Content Data Store Analysis – An analysis on storing the CMS contentsin JCR based File System and RDBMS to determine scalability in terms of data storage and retrieval. Each data model supports a philosophy, to structure and access data. On the one hand, the success of the relational model comes in large part from the facilities which are offered to describe clear data structures. On the other hand, the success of the JCR specification relates essentially to the facilities which are offered to express flexible data structures.Comparison results in Table 1.

Table 1: **Comparison of Relational Database& JCR File System (No-SQL)**

S. NO	**LEVELS OF COMPARISON**	**JCR File System (No-SQL)**	**RDBMS**
1	**DATA MODEL LEVEL**		
	Structure	Unstructured Semi structured Structured	Structured
	Integrity	Entity integrity Domain integrity Referential integrity Transitive integrity in hierarchies	Entity integrity Domain integrity Referential integrity Tools to manage data coherency
	Operations and Queries	Selection Equi-join operations Full text search operation Transitive queries on hierarchies	Selection Projection Rename Join operations Domain operation Create, read, update, delete statements
	Navigation	Navigation API Traversal access Direct access Write access	Not supported
2	**SPECIFICATION LEVEL**		
	Inheritance	Node types inheritance Node inheritance	Not supported
	Access Control	Record level	Table and Column level Record level not supported
	Observation	Record level Un-persisted event listeners Application interaction supported	Table level Persisted triggers Application interaction not supported
	Version Control	Supported	Not Supported
3	**PROJECT LEVEL**		
	Schema Understandability	Data Guides or Graphs Summarize the architecture Not impacted by many-to-many associations	Entity Relationship Represent the whole architecture Impacted by many-to-many associations
	Code complexity	Simple for Navigation Complex for Operations	Complex for Navigation Simple for Operations
	Changeability	More agile Decoupled from the application	More rigid Coupled with the application

Comparison of No-SQL Databases: The No-SQL (non-relational) database features elasticity and scalability in combination with a capability to store big data and work with cloud computing systems. No-SQL data management systems are inherently schema-free and eventually consistent

(complying with BASE rather than ACID). They have a simple API, serve huge amounts of data and provide high throughput.
An analysis and evaluation is done on some of the key and popular No-SQL solutions, Reference Section 7[x]

- **Hadoop** – a column-oriented and key-value store
- **Cassandra** – a column family store
- **CouchDB** – a document store
- **MongoDB** – a document-oriented database
- **Riak** – a key value store

On analysis and evaluating the performance of the system with the above No-SQL data stores under different workloads like,

- Workload A: **Update heavily**
- Workload B: **Read mostly**
- Workload C: **Read only**
- Workload D: **Read latest**
- Workload E: **Scan short ranges**
- Workload F: **Read-Modify-write**
- Workload G: **Write heavily**

Each workload is being defined by:
1) The number of records manipulated (read or written)
2) The number of columns per each record
3) The total size of a record or the size of each column
4) The number of threads used to load the system

Configuration settings for each type of the workloads are,

1) 100,000,000 records manipulated
2) The total size of a record equal to 1Kb
3) 10 fields of 100 bytes each per record
4) Multithreaded communications with the system (100 threads)

No-SQL DatabasesComparison Result: Performance of each No-SQL database under different types of loads is recorded with respect to Average Latency (in milliseconds) and Throughput (in operations/sec). Hadoop based HBase demonstrated by far the best writing speed. With pre-created regions and deferred log flush enabled, it reached 40K ops/sec. Cassandra also showed great performance during the loading phase with around 15K ops/sec.Hence on comparing various parameters and performing various tests **Apache Hadoop** is chosen as an optimal solution to obtain maximum scalability.
Hadoop Big Data: Hadoop consists of two major components,

- **HDFS** – The file store is called Hadoop Distributed File System. HDFS provides scalable, fault tolerant storage at low cost. The HDFS software detects and compensates for hardware issues, including disk problems and server failure. It is the primary storage system used by Hadoop applications. HDFS creates multiple replicas of data blocks and distributes them on compute nodes throughout a cluster to enable reliable, extremely rapid computations.
- **Map Reduce** – The second major component of Hadoop is the parallel data processing system called Map Reduce. The Map Reduce function is operated on a set of key, value pairs where Map is applied in parallel on input data set which produces output keys and list of values for each key depending upon the functionality. Reduce is then applied in parallel to each group again producing a collection of key, values. Hence the Mapper output partitioned per reducer is equal to No. of reduce task for that job.

5. Benchmark Hadoop Document Management System on an Oracle Big Data Appliance

At Tata Consultancy Services (TCS) the coreCMS is configured to use Hadoop Data Store using a plugin. Refer Section 7[i] for the patent filled on this CMS plugin. A Hadoop based Document Management System (DMS) is developed from the core CMS.

A Proof of Concept (POC) is executed to benchmark Hadoop DMS on Oracle Big Data Appliance (BDA) at the facility provided by Oracle in their lab. POC is executedto study the feasibility of using an Oracle BDA as the consolidated platform for uploading, storing, indexing and sorting Meta-tags for the document management purposes. Reference Section 7[ii].

The primary aim is to measure capacity & scalability of Hadoop DMS incorporating Oracle's BDA in following phases:
Phase 1: Entails loading **5 million documents** in BDA and to retrieve them based on search criteria within stipulated Service Level Agreement (SLA) for this operation of searching through Meta-tags is **3 seconds**.
Phase 2: Entails loading **10 million documents** in BDA and to retrieve them based on search criteria within stipulated SLA for this operation of searching through Meta-tags is **8 seconds**.

Performance demonstration of executing use cases shall be recorded using **JMETER**.
The Use Case Scenarios considered are,

- Upload Document
- Search Document
- Retrieve Document
- Remove document
- Upload, Search & Retrieve document

Hardware & Software Specification used for POC

- Big Data appliance (BDA) has 18 Nodes in total and each node has 24 cores of 3.07 GHz speed
- 216 Intel® Xeon® E5 Processors in total
- 864 GB total memory with each node has 48 GB
- 648 TB total raw storage capacity with each node having 36 TB
- 40Gb/sec InfiniBand Network
- 10Gb/sec Data Center Connectivity
- Oracle Linux 5.6
- Java Hotspot VM
- Cloudera CDH 4.1.2
- Hadoop
- HBASE 0.92 CDH4u2
- Cloudera Manager
- Java SDK 1.6

TCS environment

- 1 Name Node + 4 Data Nodes
- Total: 48 GB: 12 GB RAM on each node
- HDD Size: 1.6TB
- RHEL 5
- Hadoop: Cloudera 0.20.2-cdh3u5
- HBase: Cloudera 0.90.6-cdh3u4

Results Summary

The result summary on use cases tested and recorded with varying concurrency and varying time scale for both 5 million and 10 million documents.

- Uploading of documents in HDFS of approximately 9.6 TB of data using native Hadoop and HBase utilities through scripts provided by TCS for 5 million documents in Phase 1and loading up to 20 TB of data for 10 million documents in Phase 2.
- Searching the documents based on search criteria stored in HBase tables using HBase REST API and removing them from HDFS for archiving using Hadoop Java API.
- Searching the documents based on search criteria stored in HBase tables and removing them from HDFS for archiving using HBase REST API.
- Hybrid combined queries are executed to search, retrieve and upload documents in HDFS.

Table 2: POC Results for Use Cases executed in Hadoop Content Management System

Use Case Type	**Description**	**5 Million Documents** (*SLA Required 3 Seconds)	**10 Million Documents**(* SLA Required 8 Seconds)
		SLA Achieved	**SLA Achieved**
Upload Document	9 Requests fired concurrently for 10 Users for 5 Minutes	1.01 Seconds	0.80 Seconds
Upload Document	9 Requests fired concurrently for 10 Users for 15 Minutes	1.11 Seconds	0.76 Seconds
Search Document	5 Attribute search fired for 100 & 200 concurrent users for 5 Minutes	0.33 Seconds	3.2 Seconds
Search Document	5 Attribute search fired for 100 & 200 concurrent users for 15 Minutes	0.22 Seconds	2.34 Seconds
Retrieve Document	Retrieve Document fired for 20 concurrent users for 5 & 15 Minutes	0.015 Seconds	1.48 Seconds
Remove Document	Remove Document fired for 10 concurrent users for 500 Iterations	0.008 Seconds	0.24 Seconds
Upload, Search & Retrieve Document	Running Hybrid queries for multiple use cases with different concurrency of users for 5 Mins	0.69 Seconds	4.2 Seconds
Upload, Search & Retrieve Document	Running Hybrid queries for multiple use cases with different concurrency of users for 15 Mins	0.43 Seconds	4.08 Seconds

The POC convincingly conclude that TCS Hadoop DMS on Oracle BDA has performed beyond set expectations. BDA's performance was at least 3 times the desired service level agreement (SLA), best being approximately 100X. The observed numbers in the result provides insightful learning about the performance gain BDA brings to Hadoop DMS. The CMS application shall be significantly augmented when integrated on BDA. CMS application is designed to perform for a larger set of data, and an engineered solution will provide scalable and durable platform to deliver high performance.

6. Conclusion

A detailed research on CMS solutions and the technology evaluation is done in terms of various parameters. The product line approachis adopted in architecting and developing the CMS based on enterprise needs by wiring the required CMS components. The Scalability of the CMS is evaluated in terms of different types of data repository and data store for storing big and growing content. The Big Data technologies are compared on specific parameters and selected. The selected technology is evaluated and benchmarked with Oracle BDA to assure a Scalable Content Management System.

7. References

i. ***Patent Reference** - **India** - 2097/MUM/2013, **US** - 14/027137, **Europe** -EP131840*

ii. ***POC Document on Benchmarking Hadoop DMS on Oracle Big Data Appliance** published in TCS and Oracle.*

iii. *http://open-source.gbdirect.co.uk/migration/benefit.html#reliability*

iv. *http://quintagroup.com/cms/open-source*

v. *http://gigaom.com/2013/03/17/dont-blame-security-breaches-on-open-source-technology-the-problem-is-lack-of-oversight/*

vi. *http://www.javaworld.com/javaworld/jw-04-2005/jw-0411-spring.html*

vii. *http://highscalability.com/guerrilla-capacity-planning-and-law-universal-scalability*

viii. *http://www.perfdynamics.com/Manifesto/USLscalability.html*

ix. *http://www.slideshare.net/outerthought/nosql-with-hadoop-and-hbase*

x. *http://www.networkworld.com/news/tech/2012/102212-nosql-263595.html?page=1*

xi. *http://db-engines.com/en/system/Cassandra%3BHBase%3BMongoDB*